



# Desarrollo de aplicaciones multiplataforma (DAM)

## LagVis

AUTOR: SANTIAGO PÉREZ DÍAZ-RUBÍN  
TUTOR: ALEJANDRO JÍMENEZ VITORIA  
2º DAM JUNIO 2025  
ENLACE A CÓDIGO FUENTE PROYECTO

## ÍNDICE

### Contenido

|   |    |
|---|----|
| Contenido .....   | 2  |
| 1. Introducción: .....  | 3  |
| 2. Motivación:.....   | 4  |
| 3. Resumen (Abstract) .....                                       | 5  |
| 4. Objetivos del proyecto: .....                                  | 5  |
| 5. Metodología: .....   | 7  |
| 6. Tecnologías y herramientas utilizadas .....                    | 9  |
| 7. Planificación.....   | 11 |
| 8. Análisis .....   | 13 |
| 9. Diseño.....  | 24 |
| 10. Instalación, ejecución, configuración y puesta en marcha..... | 27 |
| 11. Conclusiones .....  | 32 |
| 12. Vías Futuras .....  | 35 |
| 13. Glosario: .....   | 37 |
| 14. WEBGRAFÍA .....   | 40 |

## 1. Introducción:

Justificación del proyecto:

En España, muchas personas desconocen cuáles son sus derechos laborales, tanto trabajadores como empleadores. Esto puede generar conflictos, malentendidos y, en algunos casos, situaciones injustas que podrían evitarse si se tuviera acceso a la información de una forma clara y sencilla. Por ejemplo, hay trabajadores que no saben si están cobrando lo que les corresponde, o empresas que sin querer pueden cometer errores administrativos por no estar bien informadas.

Uno de los problemas más comunes es que los convenios colectivos, que regulan gran parte de las condiciones laborales según cada sector, suelen ser difíciles de entender. La mayoría de la gente no está familiarizada con el lenguaje técnico que se utiliza en ellos, y eso complica mucho el acceso real a sus derechos. Esta falta de comprensión afecta tanto a empleados como a empresarios, y puede acabar generando tensiones innecesarias en el entorno laboral.

Por eso, este proyecto busca ofrecer una solución tecnológica que simplifique ese acceso. **LagVis** es una aplicación móvil pensada para que cualquier persona pueda consultar de forma rápida y fácil información sobre salarios, vacaciones, permisos, y otros aspectos relacionados con su trabajo. La idea es que la app no solo muestre datos, sino que los explique con ejemplos y un diseño intuitivo, adaptado a todo tipo de usuarios.

Además, como desarrollador, me pareció interesante afrontar un proyecto que no solo tuviera una parte técnica, sino también un impacto social. Me permite aplicar todo lo aprendido durante el ciclo (desde programación y diseño hasta organización del trabajo en equipo), al mismo tiempo que desarrollo algo que puede ser útil para mucha gente.

Por todo esto, considero que **LagVis** es un proyecto que merece la pena llevar a cabo: por su utilidad, por el valor que puede aportar a los usuarios, y por todo lo que representa como reto personal y profesional.

Breve resumen del trabajo:

Este proyecto consiste en el desarrollo de **LagVis**, una aplicación móvil pensada para facilitar el acceso a la información laboral en España. La idea nace a raíz de un problema bastante común: muchas personas no tienen claro cuáles son sus derechos

---

---

en el trabajo, y eso puede llevar a errores, injusticias o conflictos innecesarios entre trabajadores y empleadores.

Con esta app, se busca ofrecer una solución sencilla, visual y fácil de usar, donde cualquier usuario pueda consultar aspectos clave como su convenio colectivo, el salario mínimo del sector, los días de vacaciones o qué hacer en caso de despido. Todo esto presentado con un diseño intuitivo, resúmenes comprensibles y ejemplos prácticos que ayuden a entender mejor la normativa.

Durante el desarrollo del proyecto, se han aplicado herramientas y tecnologías como **Firebase** y se ha seguido una metodología ágil (Scrum) para organizar el trabajo por fases. Además, se ha prestado especial atención a la experiencia de usuario (UX) y al diseño visual de la aplicación.

El objetivo final es crear un producto útil, accesible y que pueda marcar una diferencia real en el día a día de las personas, ayudándolas a conocer y defender mejor sus derechos laborales.

## 2. Motivación:

La idea de este proyecto surgió a partir de mi propia experiencia personal. Durante un tiempo estuve trabajando sin tener muy claro cuáles eran mis derechos laborales. Intenté buscar información por mi cuenta, pero cada vez que me enfrentaba a los convenios colectivos o a la normativa laboral, todo me parecía demasiado complicado. El lenguaje técnico, los documentos extensos y la falta de explicaciones claras hacían que acabara rindiéndome y dejando el tema de lado.

Tiempo después, hablando con una amiga que estudia un ciclo superior de Administración y Finanzas, me comentó que le rondaba una idea: crear una aplicación que hiciera más fácil entender este tipo de información. A ella siempre le había interesado el tema de los derechos laborales, y le parecía útil desarrollar una herramienta que ayudara tanto a trabajadores como a empresas a tener más claro lo que dice la ley. Me dijo que necesitaba a alguien que supiera programar para poder hacer realidad esa idea, y no dudé en ofrecerme.

Me pareció una oportunidad perfecta para unir lo que he aprendido en desarrollo de aplicaciones con un proyecto que, además de ser interesante a nivel técnico, puede tener un impacto real en la vida de muchas personas. Por eso decidí convertir esta idea en mi proyecto final: **LagVis**, una app pensada para informar de forma clara, accesible y útil sobre derechos laborales y convenios colectivos.

---

---

### 3. Resumen (Abstract)

LagVis is a mobile application designed to help users in Spain easily understand their labor rights and collective agreements. It aims to simplify complex legal information, making it accessible to both employees and employers.

The application allows users to look up information relevant to their work contracts, such as salary expectations, official leave days, and procedures related to dismissals. It includes practical tools like calculators to estimate severance pay and dismissal indemnities. Users can also stay updated with the latest news on labor legislation and save articles of interest. Additionally, the app provides access to manage user profile information and offers a direct link to the official Spanish "Vida Laboral" portal to check employment history.

Ultimately, LagVis seeks to empower users with knowledge about their labor rights and obligations, aiming to foster more transparent workplace relations and reduce potential conflicts

### 4. Objetivos del proyecto:

El objetivo principal de esta aplicación es que sea una herramienta útil para muchísimas personas en España. Queremos que cualquier persona, tanto trabajadores como jefes, puedan entender fácilmente sus derechos en el trabajo y las reglas que se aplican a su sector, que son los "convenios colectivos".

Imagina que es como tener una guía sencilla y al alcance de tu mano que te ayuda a saber qué te corresponde en tu empleo. Hoy en día, mucha gente no sabe cuánto debería ganar, cuántos días libres tiene o qué hacer si la despiden. Esto no solo afecta a los trabajadores, sino que a veces los jefes tampoco conocen bien todas las normas y pueden cometer errores sin querer. La falta de este conocimiento genera muchos malentendidos y problemas en el trabajo. Por ejemplo, a veces los trabajadores cobran menos de lo que deberían o no les dan los permisos que les corresponden. Y para las empresas, esto puede significar perder tiempo y dinero en conflictos que podrían haberse evitado. De hecho, se estima que más del 60% de las personas en España no entienden del todo los convenios colectivos.

Por eso, con esta aplicación, queremos hacer que toda esa información complicada sea "súper fácil" de encontrar. La idea es que, de una forma rápida y sencilla, sin tener que complicarse la vida con textos legales difíciles, cualquier persona pueda saber a qué atenerse en su trabajo. Así, se busca evitar problemas y conseguir que las relaciones laborales sean más justas y tranquilas para todos.

#### Objetivos Específicos del Proyecto

Para lograr el gran objetivo de la aplicación, nos hemos puesto varias metas más pequeñas y concretas:

- **Crear un sistema para registrarse y entrar a la aplicación de forma segura:** Queremos que cada usuario tenga su propia cuenta personalizada, lo
-

que le permitirá personalizar la experiencia según su sector laboral. Esto significa que podrás registrarte con tu correo y una contraseña, y así la aplicación podrá ofrecerte una experiencia más a tu medida.

- **Diseñar una forma eficaz de buscar convenios colectivos:** La aplicación te permitirá encontrar el convenio exacto que necesitas. Podrás buscarlo filtrando por el tipo de trabajo que tienes (tu sector laboral) y por la ciudad o región donde vives (comunidad autónoma). Así no tendrás que leer convenios que no te afectan.
- **Hacer que los convenios se vean de forma clara y fácil de entender:** Una vez que encuentres tu convenio, la aplicación te mostrará un resumen sencillo de los puntos más importantes. No queremos que te marees con lenguaje legal. Verás la información clave, como el salario mínimo que te corresponde, cuántos días de vacaciones tienes o qué permisos laborales existen para tu caso.
- **Tener una sección de noticias laborales siempre al día:** La aplicación incluirá un espacio donde podrás ver las últimas noticias y novedades importantes sobre leyes y normativas laborales. Así estarás siempre informado sobre cualquier cambio que pueda afectarte en el trabajo.
- **Integrar calculadoras para facilitar gestiones:** Queremos que la aplicación sea una herramienta muy práctica. Por eso, incluirá calculadoras que te ayudarán a hacer cuentas importantes. Una de ellas será para que los trabajadores puedan calcular cuánto les corresponde de indemnización si les despiden. La otra calculadora será para los jefes, que les permitirá estimar cuánto tendrían que pagar si quieren despedir a un empleado, considerando todas las opciones legales. Estas calculadoras simplifican cálculos que a menudo son confusos para los usuarios.
- **Ofrecer acceso rápido a herramientas oficiales del estado:** Para complementar la información y calculadoras, la aplicación pondrá a tu alcance enlaces directos a las herramientas y recursos más útiles que ofrecen los organismos oficiales del estado relacionados con el ámbito laboral. Así, si necesitas algo más específico o verificar información oficial, lo tendrás a mano.
- **Diseñar una aplicación muy fácil de usar:** Es crucial que la aplicación sea intuitiva, es decir, que sepas cómo usarla casi sin pensar. Queremos una interfaz (la parte que ves y con la que interactúas) sencilla y clara, con botones grandes y colores que contrasten, para que navegar y encontrar lo que buscas sea muy rápido y cómodo en cualquier dispositivo móvil.

- **Asegurarse de que la aplicación funciona perfectamente en todos los teléfonos Android:** Antes de que la aplicación esté lista, se harán muchas pruebas para garantizar que funciona bien en diferentes modelos de teléfonos Android y con distintas resoluciones de pantalla. Queremos que la experiencia sea fluida y sin fallos para todos los usuarios.
- **Crear un sistema de avisos y notificaciones:** En el futuro, la aplicación podría enviarte alertas importantes. Por ejemplo, si hay un cambio en tu convenio colectivo o en alguna normativa laboral que te afecte directamente, la aplicación te avisará para que estés al tanto en tiempo real

## 5. Metodología:

Para la gestión y el desarrollo de este proyecto, he optado por implementar la metodología **Scrum**. Esta elección no es aleatoria, sino que responde a una serie de ventajas fundamentales que Scrum ofrece, especialmente en el contexto de un proyecto de desarrollo de software como el que he llevado a cabo.

### ¿Por qué he elegido Scrum?

La principal razón para decantarme por Scrum es su naturaleza **ágil**. La agilidad es crucial en el desarrollo de software, ya que me permite ser flexible y adaptarme rápidamente a los cambios que puedan surgir a lo largo del proceso. En un proyecto como el mío, donde las funcionalidades pueden evolucionar (como la inclusión de nuevas calculadoras o la integración de herramientas estatales), o donde el diseño de la interfaz puede requerir ajustes basados en la retroalimentación o en mi propia experiencia durante el desarrollo, la capacidad de adaptación de Scrum es invaluable. Me permite pivotar, ajustar prioridades y refinar la solución sin que esto suponga un retraso significativo o una ruptura en el flujo de trabajo.

Además, Scrum estructura el trabajo en períodos cortos y definidos, conocidos como "**sprints**". En mi caso, cada sprint tiene una duración de **dos semanas**. Esta cadencia me proporciona un mecanismo de control y seguimiento constante del progreso. Al finalizar cada sprint, tengo la oportunidad de revisar de forma exhaustiva lo que se ha logrado, evaluar si los objetivos propuestos para ese ciclo se han cumplido, y ajustar las prioridades para el siguiente sprint. Esto es esencial para mantener el proyecto enfocado, evitar desviaciones y asegurar que las tareas más importantes se completen de manera eficiente en el plazo establecido.

Aunque soy el principal desarrollador de este proyecto, los principios de Scrum, como la colaboración constante y la retroalimentación, se han internalizado. La "revisión del sprint" es mi momento para auto-evaluar el avance y definir mejoras, aplicando así una retroalimentación continua que impulsa la calidad del producto.

---

---

En resumen, la elección de Scrum se justifica por:

- Su capacidad para **adaptarse a los cambios**.
- La posibilidad de **controlar el progreso** en ciclos cortos y frecuentes.
- La garantía de que las **tareas prioritarias se completen de forma eficiente**.
- El fomento de un **flujo de trabajo organizado y de mejora continua**.

### Fases de Desarrollo del Proyecto bajo el Modelo Scrum

El desarrollo de la aplicación se ha estructurado en una serie de "sprints" de dos semanas, siguiendo un ciclo iterativo y incremental. Las fases clave dentro de cada sprint son las siguientes:

1. **Planificación del Sprint (Sprint Planning)**: Al inicio de cada sprint, he definido claramente las tareas específicas a realizar durante las próximas dos semanas. Esto incluye establecer objetivos claros para el sprint. Por ejemplo, en el primer sprint, los objetivos se centraron en "crear la estructura básica de la aplicación y las funcionalidades esenciales". Las tareas iniciales incluyeron el diseño de la base de datos en Firebase, la creación de la pantalla de registro de usuarios, la implementación de la funcionalidad de inicio de sesión, y la configuración del repositorio del proyecto en GitHub.
2. **Ejecución del Sprint (Sprint Execution)**: Durante las dos semanas del sprint, he trabajado activamente en las tareas definidas. Para mantener la eficiencia y evitar bloqueos, me he enfocado en una o dos tareas simultáneamente. Diariamente, he realizado una revisión rápida del progreso (similar a un *Daily Scrum*), lo que me ha permitido evaluar el avance y ajustar el enfoque si era necesario. Las tareas han avanzado a través de un tablero visual (tipo Kanban), pasando de "Pendiente" a "En progreso" y finalmente a "Completada".
3. **Revisión del Sprint (Sprint Review)**: Al finalizar cada sprint, he llevado a cabo una revisión exhaustiva para evaluar los avances realizados y determinar si los objetivos del sprint se han cumplido. Por ejemplo, en una revisión, pude constatar que la "pantalla de inicio de sesión y funcionalidad básica de registro" estaban completadas, mientras que "finalizar la integración de Firebase con la base de datos" seguía pendiente. También he identificado posibles mejoras, como "ajustar el diseño para que sea compatible con dispositivos de baja resolución".
4. **Priorización para el Siguiente Sprint (Next Sprint Planning)**: Basándome en los resultados y las lecciones aprendidas del sprint actual, he planificado y priorizado las tareas para el siguiente ciclo de desarrollo, ajustándolas según las necesidades actuales del proyecto. Un ejemplo de tareas para un segundo

sprint podría incluir la "implementación de la funcionalidad de búsqueda de convenios", el "diseño de la pantalla de resultados de búsqueda" y la "creación de un prototipo funcional de la calculadora de liquidaciones".

Para la implementación de este modelo, he utilizado **Firebase** para la integración de la base de datos y la autenticación de usuarios, y **GitHub** para el control de versiones y la gestión del código, lo que asegura un flujo de trabajo organizado y la posibilidad de seguir los cambios realizados en el proyecto.

## 6. Tecnologías y herramientas utilizadas

Para construir esta aplicación, he elegido cuidadosamente varias tecnologías y herramientas. Cada una fue seleccionada por lo que aportaba al proyecto, buscando que la aplicación fuera sólida, eficiente, escalable y, sobre todo, fácil y agradable de usar.

### 1. El Entorno de Desarrollo y el Lenguaje Principal: Android Studio y Java

- **Android Studio:** Este es el software principal que he usado para desarrollar la aplicación. Es el entorno de desarrollo oficial de Google para Android, y me ha proporcionado todas las herramientas necesarias: desde diseñar las pantallas hasta probar el código y gestionar el proyecto completo. Es un "taller" muy potente y completo para crear apps.
- **Java:** Este es el lenguaje de programación fundamental que he empleado para la lógica de la aplicación. Lo elegí porque es el lenguaje con el que **hemos trabajado y adquirido experiencia durante estos dos años de Grado Superior**. Por ello, me siento cómodo y familiarizado con su sintaxis y estructura, lo que ha facilitado la implementación de las funciones principales de la app. Además, al ser muy usado en Android, hay una gran comunidad y recursos disponibles.

### 2. La Creación de las Pantallas: XML

- **XML (eXtensible Markup Language):** Para diseñar cómo se ven las pantallas de la aplicación (dónde va cada botón, cada texto o cada imagen), he utilizado XML. En el desarrollo Android, es el método estándar para definir la estructura visual. Lo elegí porque:
  - Me permite separar claramente el diseño de lo que se ve (el XML) de la parte que hace que la app funcione (el código Java). Esto hace que el proyecto esté más ordenado y sea más fácil de mantener.

- Facilita el diseño, ya que Android Studio tiene una herramienta visual que me permite ver en tiempo real cómo queda cada pantalla mientras la construyo.
- Ayuda a que la interfaz de la app se adapte bien a los diferentes tamaños y resoluciones de pantalla de los móviles de hoy en día, lo cual es vital para una buena experiencia de usuario.

### 3. La Gestión de Usuarios y Datos Básicos: Firebase

- **Firebase:** Esta es una plataforma de Google que ha sido crucial para la "parte de atrás" de la aplicación, es decir, para la gestión de usuarios (que puedan registrarse y entrar) y para guardar ciertos datos. La elegí por:
  - Su facilidad de integración con Android Studio y Java, lo que simplifica mucho el trabajo.
  - Ofrece un sistema de **autenticación** robusto y fácil de usar, lo que es esencial para que cada usuario tenga su cuenta personalizada y segura.
  - Dispone de una base de datos que permite guardar información y sincronizarla casi al instante, lo que es útil para datos que cambian a menudo, como las noticias.
  - Es un servicio gestionado por Google, lo que significa que puede crecer automáticamente si la aplicación gana muchos usuarios, sin que yo tenga que preocuparme por la infraestructura de servidores.

### 4. La Base de Datos Avanzada y la Conexión con la App: Google Cloud (SQL Cloud y Cloud Run)

- **Google Cloud SQL (para MySQL):** Además de Firebase, decidí usar una base de datos relacional más potente, MySQL, alojada en Google Cloud. Esta elección se debe a que, para manejar datos más estructurados y complejos como todos los convenios colectivos, MySQL ofrece mayor flexibilidad para hacer búsquedas avanzadas y asegurar que los datos estén bien organizados.  
Al estar en Google Cloud:
  - Google se encarga del mantenimiento de la base de datos (copias de seguridad, actualizaciones), lo que me permite centrarme más en el desarrollo de la app.
  - Es un servicio muy fiable y puede expandirse según las necesidades del proyecto.

- **Google Cloud Run:** Esta herramienta es fundamental porque permite que la aplicación móvil se comunique de forma segura y eficiente con la base de datos MySQL en la nube. Cloud Run me permite desplegar un pequeño programa que se activa solo cuando la app lo necesita. Esto significa que:
  - Los recursos se ajustan automáticamente a la demanda, lo que optimiza el coste.
  - Actúa como un intermediario seguro (una API) al que la app móvil le pide información o le envía datos para guardar, sin conectarse directamente a la base de datos. Esto centraliza la lógica y protege los datos.

## 5. El Control de Versiones del Código: GitHub

- **GitHub:** Esta plataforma ha sido una herramienta indispensable para llevar un control de todas las versiones de mi código. La elegí porque:
  - Me permite tener un historial detallado de cada cambio que hago en el código. Si cometo un error, puedo volver fácilmente a una versión anterior.
  - Aunque el proyecto es individual, fomenta buenas prácticas de desarrollo y organización del código, como la creación de "ramas" para trabajar en nuevas funcionalidades.
  - Sirve como un respaldo seguro para todo mi código en la nube, evitando pérdidas accidentales de trabajo.

La combinación estratégica de estas tecnologías, desde el entorno de desarrollo y el lenguaje principal, pasando por las herramientas para la interfaz y la gestión de datos, hasta la infraestructura de backend y el control de versiones, ha sido clave para construir una aplicación funcional, bien organizada y con capacidad para crecer en el futuro.

## 7. Planificación

### Estimación de Recursos

#### Tiempo

El desarrollo del proyecto se ha llevado a cabo a lo largo de aproximadamente 6 meses, distribuidos en fases de trabajo estructuradas mediante la metodología ágil

---

---

Scrum, en ciclos de sprints de dos semanas. La planificación temporal ha sido la siguiente:

- **Definición de requisitos:** 2 semanas (completada).
- **Diseño de arquitectura e interfaz:** 3 semanas (completada).
- **Desarrollo funcional (registro, login, búsqueda de convenios):** 6 semanas (completado).
- **Integración de funcionalidades (noticias, perfil, ajustes):** 4 semanas (completado).
- **Pruebas, correcciones y mejoras:** 4 semanas (completado).
- **Documentación y entrega final:** 2 semanas (en curso, finaliza el 09/06/2025).

### Costes

Aunque el proyecto no ha supuesto gastos económicos directos gracias al uso de herramientas gratuitas, se puede estimar un coste aproximado en función del tiempo invertido:

- **Horas estimadas de trabajo total:** ~220 h.
- **Valoración aproximada (10 €/h):** ~2.200 €.
- **Costes adicionales (hardware, conexión, pruebas en móvil):** ~300 €.
- **Coste total estimado: 2.500 € aprox.**

### Personal

Todo el desarrollo ha sido realizado de manera individual, asumiendo los roles de desarrollador, diseñador, Scrum Master y tester. Esta organización ha permitido un control total sobre el proyecto, aunque ha requerido una planificación detallada y constancia para cumplir los plazos.

| Fase del Proyecto                      | Fecha Estimada             | Fecha Real                 | Estado     |
|--|----------------------------|----------------------------|------------|
| Definición de Requisitos               | 07/11/2024 –<br>20/11/2024 | 07/11/2024 –<br>20/11/2024 | Completado |
| Diseño de arquitectura e interfaz      | 21/11/2024 –<br>12/12/2024 | 28/10/2024 –<br>15/11/2024 | Completado |
| Desarrollo funcional (Login, Registro) | 13/12/2024 –<br>31/01/2025 | 01/12/2024 –<br>31/01/2025 | Completado |
| Búsqueda de convenios y noticias       | 01/02/2025 –<br>01/03/2025 | 01/02/2025 –<br>28/02/2025 | Completado |
| Integración, pruebas y mejoras         | 02/03/2025 –<br>30/03/2025 | 02/03/2025 –<br>29/03/2025 | Completado |
| Documentación y entrega final          | 01/04/2025 –<br>15/04/2025 | 25/05/2025 –<br>09/06/2025 | En curso   |

## 8. Análisis

### Requisitos Funcionales

Los requisitos funcionales definen las funcionalidades que **LagVis** ofrece actualmente. Todos ellos han sido implementados y probados con éxito.

#### 1. Registro de usuario

Permite a los nuevos usuarios crear una cuenta introduciendo sus datos personales y laborales (nombre, correo electrónico, sector laboral, comunidad autónoma, etc.).

#### 2. Inicio de sesión

Los usuarios registrados pueden acceder a la aplicación introduciendo su correo electrónico y contraseña.

#### 3. Búsqueda de convenios colectivos

El sistema permite filtrar convenios según sector laboral y comunidad autónoma, mostrando únicamente la información relevante.

#### 4. Visualización de convenios

Cada convenio encontrado se presenta en formato resumido e intuitivo, destacando datos clave como salario mínimo, días libres y derechos básicos.

#### 5. Lectura de noticias laborales

Se incluye una sección con noticias de actualidad sobre el entorno laboral. El usuario puede deslizar entre ellas y acceder a una vista ampliada con el contenido completo.

#### 6. Visualización del perfil de usuario

El usuario puede acceder a su perfil para consultar sus datos personales y laborales. Por decisión de diseño, **no se permite editar la información desde la aplicación**.

#### 7. Calculadora de finiquito y despido

Se han integrado herramientas para calcular finiquitos e indemnizaciones por despido de forma sencilla, rápida y precisa, adaptadas al marco legal español.

---

---

**8. Acceso directo a la página oficial de vida laboral**

La app incluye una funcionalidad que redirige al usuario a la web oficial de la Seguridad Social para consultar su informe de vida laboral.

## Requisitos No Funcionales

Estos requisitos establecen cómo debe comportarse la aplicación en cuanto a rendimiento, seguridad, usabilidad y calidad general.

### 1. Usabilidad

La aplicación ha sido diseñada para ser intuitiva y accesible para todo tipo de usuarios, incluso aquellos sin experiencia en tecnología o leyes laborales.

### 2. Compatibilidad

LagVis funciona correctamente en dispositivos Android con versiones 8.0 (Oreo) en adelante, y está optimizada para diferentes tamaños de pantalla.

### 3. Rendimiento

La carga de convenios, noticias y otras funciones responde en menos de 2 segundos, asegurando una experiencia fluida.

### 4. Seguridad

El acceso y almacenamiento de datos de usuarios está gestionado mediante Firebase Authentication, respetando buenas prácticas de protección de datos.

### 5. Escalabilidad

El diseño modular de la aplicación permite añadir nuevas funcionalidades en el futuro sin necesidad de reestructurar completamente la base del proyecto.

### 6. Mantenibilidad

El código fuente está organizado en módulos bien diferenciados, lo que facilita el mantenimiento, revisión y depuración del sistema.

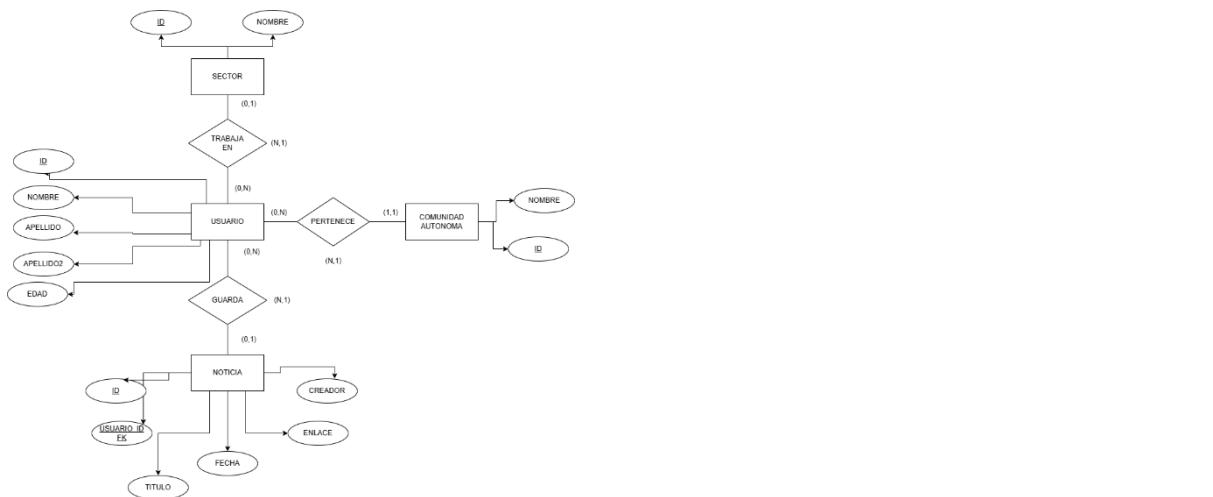
### 7. Disponibilidad

Gracias al uso de servicios en la nube como Firebase, se garantiza una alta disponibilidad del sistema, minimizando posibles caídas o interrupciones.

---

---

## Diagrama de Entidad Relación:



## Entidades y Atributos

### 1. Usuarios

- id (PK)
- uid (UNIQUE)
- nombre
- apellido
- apellido2
- edad
- sector\_id (FK → Sectores)
- comunidad\_id (FK → Comunidades)

### 2. Sectores

- id (PK)
- nombre (UNIQUE)

### 3. Comunidades

- id (PK)
- nombre (UNIQUE)

### 4. Noticias\_guardadas

- id (PK)
- usuario\_id (FK → Usuarios.uid)
- titulo
- enlace
- fecha
- creador

### 1. Relación Usuarios – Sectores

- Tipo: Muchos usuarios pueden pertenecer a un solo sector
  - Cardinalidad: (0,N) Usuarios — (0,1) Sectores
  - Representación: Un usuario puede estar vinculado a uno o ningún sector; un sector puede tener muchos usuarios.
  - Justificación: Aunque en la vida real una persona pueda tener experiencia o desempeñarse en varios sectores, en el diseño de esta aplicación se ha optado por permitir que cada usuario esté vinculado únicamente a un sector principal. Esta decisión responde a criterios de simplicidad estructural, facilidad en las consultas y enfoque funcional de la plataforma, que prioriza ofrecer contenidos y funcionalidades relacionadas con el sector más representativo del usuario. Además, se contempla la posibilidad de que un usuario aún no haya definido su sector, por lo que la relación admite valores nulos.
- 
-

## 2. Relación Usuarios – Comunidades

- Tipo: Muchos usuarios pueden pertenecer a una sola comunidad
- Cardinalidad: (0,N) Usuarios — (0,1) Comunidades
- Representación: Un usuario puede estar vinculado a una o ninguna comunidad autónoma; una comunidad puede tener muchos usuarios.
- Justificación: Cada usuario puede estar asociado a una única comunidad autónoma, lo cual permite personalizar la experiencia en función de su ubicación geográfica o administrativa. La relación está diseñada para ser opcional, contemplando casos en los que el usuario aún no haya especificado su comunidad. Esta estructura facilita la gestión de contenidos localizados y la segmentación por regiones.

## 3. Relación Usuarios – Noticias\_guardadas

- Tipo: Un usuario puede guardar muchas noticias
- Cardinalidad: (1,N) Noticias — (1,1) Usuario
- Representación: Cada noticia guardada pertenece a un solo usuario (a través del usuario\_id), y cada usuario puede guardar múltiples noticias.
- Justificación: La relación refleja las noticias que un usuario ha decidido guardar dentro de la plataforma. Aunque las noticias no están asociadas de forma exclusiva a un único usuario, sí se registra cada instancia de guardado de manera individual. Esto significa que varios usuarios pueden guardar la misma noticia, pero cada usuario solo puede guardarla una vez, lo cual garantiza la integridad y evita duplicados. Este modelo permite construir un historial personalizado de noticias guardadas sin generar redundancia en el sistema.

## Casos de Uso

### Actores principales de la aplicación:

**Usuario registrado:** Persona que accede a LagVis mediante sus credenciales para utilizar sus funcionalidades principales.

**Usuario no registrado:** Persona que aún no tiene cuenta en LagVis y puede acceder solo a la pantalla de registro o login.

**Sistema LagVis:** Aplicación central que procesa solicitudes, gestiona datos y responde a las interacciones del usuario

### Caso de Uso: Registro de usuario

- **Objetivo:** Permitir que una nueva persona cree una cuenta en la plataforma.
- **Actor principal:** Usuario no registrado.
- **Precondición:** El usuario no debe estar registrado con el mismo correo electrónico.
- **Postcondición:** Se crea un nuevo usuario con un uid único.
- **Flujo principal:**
  - . El usuario accede a la pantalla de registro.
  - . Introduce los datos obligatorios: nombre, apellidos, edad, correo electrónico y contraseña.
  - . Opcionalmente selecciona su sector y comunidad.
  - . El sistema valida los datos (unicidad del correo, formato correcto, etc.).
  - . Se almacena el usuario en la base de datos y se genera su uid.
  - . El usuario es redirigido a la pantalla principal.

### Caso de Uso: Iniciar sesión

- **Objetivo:** Permitir a un usuario acceder a su cuenta existente.
  - **Actor principal:** Usuario registrado.
  - **Precondición:** El usuario debe haber completado el registro previamente.
- 
-

- **Postcondición:** Se crea una sesión activa.
- **Flujo principal:**
  - . El usuario accede a la pantalla de inicio de sesión.
  - . Introduce su correo y contraseña.
  - . El sistema valida las credenciales.
  - . Si son correctas, se inicia sesión y se accede a la aplicación.

### Caso de Uso: Visualizar noticias

- **Objetivo:** Permitir al usuario ver noticias una a una e indicar si le gustan o no.
- **Actor principal:** Usuario registrado.
- **Precondición:** El usuario debe haber iniciado sesión.
- **Postcondición:** El sistema registra las interacciones del usuario con las noticias.
- **Flujo principal:**
  - . El usuario accede a la sección “Ver noticias”.
  - . El sistema muestra una noticia (título, enlace, creador, fecha).
  - . El usuario tiene tres opciones: Ir a la página del periódico de la noticia y visualizar la noticia en su navegador, guardar la noticia para leerla luego o pasar a la siguiente.
  - . El sistema registra la interacción
- **Extensiones:**
  - . Si no hay más noticias disponibles, se muestra un mensaje informativo.

### Caso de Uso: Guardar una noticia

- **Objetivo:** Permitir que el usuario guarde una noticia que le ha gustado o desea consultar más adelante.
  - **Actor principal:** Usuario registrado.
  - **Precondición:** El usuario debe haber iniciado sesión y no haber guardado previamente esa misma noticia.
- 
-

- **Postcondición:** Se crea una entrada única en Noticias\_guardadas.
- **Flujo principal:**
  - . Mientras visualiza una noticia, el usuario pulsa “Guardar”.
  - . El sistema verifica que el enlace de esa noticia no esté ya guardado por el mismo usuario.
  - . Si no existe, se crea una nueva entrada en Noticias\_guardadas asociada a su uid.

### Caso de Uso: Consultar noticias guardadas

- **Objetivo:** Permitir al usuario acceder al listado de noticias que ha guardado anteriormente.
- **Actor principal:** Usuario registrado.
- **Precondición:** El usuario debe tener al menos una noticia guardada.
- **Postcondición:** Se listan todas las noticias almacenadas por ese usuario.
- **Flujo principal:**
  - . El usuario accede a la sección de “Noticias guardadas”.
  - . El sistema recupera las entradas de la tabla Noticias\_guardadas asociadas a su uid.
  - . Las noticias se presentan en formato lista con título, enlace y fecha.

### Caso de Uso: Ver su perfil de usuario

- **Objetivo:** Permitir al usuario consultar su información personal.
- **Actor principal:** Usuario registrado.
- **Precondición:** El usuario debe estar autenticado.
- **Flujo principal:**
  - . El usuario accede a su perfil.
  - . Visualiza campos como nombre, edad, sector o comunidad.

**Diagrama de clases:**

## 1. BaseActivity (Clase Base)

- Extiende: AppCompatActivity
- Métodos:
  - onCreate()
  - showCustomToast()

## 2. Actividades Principales (Extienden BaseActivity)

- LoginActivity
  - Maneja autenticación con Firebase
  - Navegación a RegisterActivity y RecoverPassword
- RegisterActivity
  - Registro de usuarios nuevos
  - Navegación a AdvancedFormRegister
- RecoverPassword
  - Recuperación de contraseña via Firebase
- MainActivity
  - Implementa: NavigationView.OnNavigationItemSelectedListener
  - Gestiona navegación entre fragmentos
- ActivityDatosGeneralesFiniquito
- ActivityResultadoDespido
- AdvancedFormRegister
- Convenio

## **Jerarquía de Fragmentos**

### 1. BaseFragment (Clase Base)

- Extiende: Fragment

### 2. Fragmentos de la Aplicación (Extienden BaseFragment o Fragment)

- FirstFragment
- SecondFragment
- ThirdFragment
- FourthFragment
- NoticiasGuardadasFragment

## **Interfaces y Clases de Datos**

### 1. NoticiasCallback (Interfaz)

- Define callbacks para manejo de noticias

### 2. NewsItem (Clase de Datos)

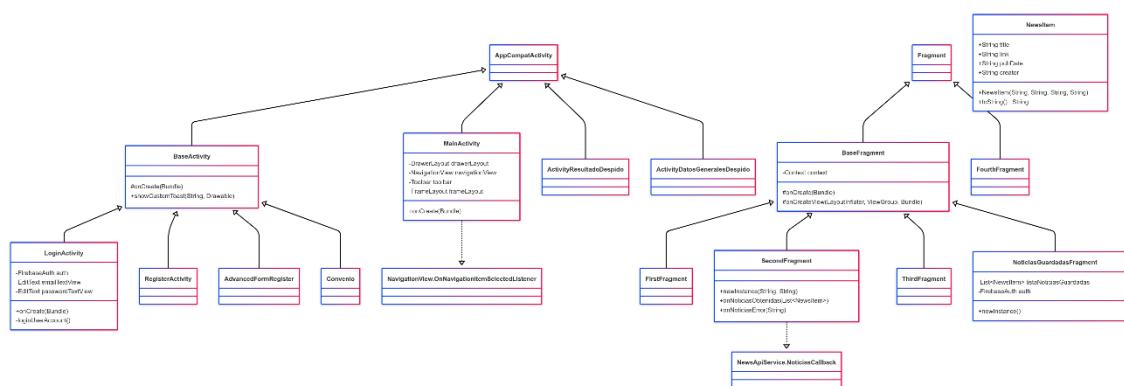
- Modelo para items de noticias

## **Sistema de Autenticación**

- Integración con FirebaseAuth
- Utilizado en:
  - LoginActivity
  - RegisterActivity
  - RecoverPassword
  - FourthFragment
  - AdvancedFormRegister

## **Relaciones Principales**

- MainActivity actúa como contenedor principal y gestiona la navegación entre fragmentos
  - Las actividades comparten funcionalidad común a través de BaseActivity
  - Los fragmentos heredan funcionalidad base de BaseFragment
  - La autenticación está centralizada usando Firebase
  - Los datos de noticias se manejan a través de NewsItem y NoticiasCallback



## 9. Diseño

## **Modelo Lógico: Diseño de la Base de Datos**

El modelo lógico representa la estructura formal de los datos que utiliza el sistema LagVis. A partir del modelo conceptual definido por entidades como Usuario, Sector, Comunidad y Noticias\_guardadas, se ha construido un esquema lógico basado en el paradigma relacional, siguiendo principios de normalización y diseño modular.

## **Tablas principales y claves:**

## 1. **Usuarios**

- id (PK, autoincremental)
  - uid (UNIQUE, identificador alternativo)
  - nombre, apellido, apellido2, edad
  - sector\_id (FK → Sectores.id)
  - comunidad\_id (FK → Comunidades.id)

## 2. Sectores

- o id (PK)

- nombre (UNIQUE)
3. **Comunidades**
- id (PK)
  - nombre (UNIQUE)
4. **Noticias\_guardadas**
- id (PK)
  - usuario\_id (FK → Usuarios.uid)
  - título, enlace, fecha, creador
  - Restricción única sobre (usuario\_id, enlace) para evitar duplicados

### Normalización:

Se han aplicado principios de **normalización hasta 3FN (Tercera Forma Normal)** para garantizar:

- Eliminación de redundancia de datos
- Integridad referencial a través de claves foráneas
- Cohesión lógica entre las entidades

Por ejemplo, los datos de sector o comunidad no se repiten en la tabla de Usuarios, sino que están en tablas independientes, vinculadas por claves foráneas. Esta separación favorece la reutilización de datos, facilita consultas estadísticas y mejora el mantenimiento del sistema.

### Justificación de Decisiones Técnicas

#### Elección del Sistema de Gestión de Bases de Datos (SGBD)

Se ha optado por utilizar MySQL, un sistema de gestión de bases de datos relacional ampliamente utilizado en entornos de producción, con un equilibrio adecuado entre robustez, rendimiento y simplicidad. Entre los factores clave de esta elección destacan:

- Su madurez tecnológica y estabilidad en proyectos web.
- Amplio soporte en plataformas de nube, incluyendo Google Cloud.

#### Despliegue en Google Cloud Platform

La base de datos MySQL se ha desplegado mediante Cloud SQL, el servicio gestionado de Google para bases de datos relacionales. Esta decisión obedece tanto a razones técnicas como estratégicas:

- Alta disponibilidad: Cloud SQL gestiona replicación automática, backups y escalabilidad.
- Seguridad: Se aprovechan las políticas de red de Google Cloud para restringir accesos y gestionar credenciales de forma segura.
- Facilidad de integración con otros servicios del ecosistema Google.

### **Scripts en PHP desplegados con Google Cloud Run**

Los scripts del backend, desarrollados en PHP, están desplegados mediante Google Cloud Run, un servicio serverless que permite ejecutar contenedores en la nube sin necesidad de gestionar servidores.

Esta arquitectura presenta múltiples ventajas:

- Escalado automático según demanda.
- Despliegue sencillo desde una imagen Docker.
- Bajo coste operativo, ya que solo se paga por el tiempo de ejecución real.
- Posibilidad de integrar fácilmente el backend con la base de datos MySQL de Cloud SQL mediante conexiones seguras gestionadas por Google.

### **Relación entre arquitectura y objetivos del proyecto**

El objetivo de LagVis es ofrecer una plataforma ligera, funcional y fácilmente desplegable, por lo que la elección de tecnologías cloud y un modelo relacional normalizado encaja perfectamente con los requisitos:

- Minimiza la carga de mantenimiento
- Asegura consistencia de datos
- Favorece el despliegue automatizado
- Permite escalar a medida que el proyecto crezca

### **Conclusión**

El modelo lógico de LagVis ha sido diseñado con criterios de integridad, escalabilidad y eficiencia. La combinación de MySQL como SGBD relacional, junto con Google Cloud SQL y Cloud Run para ejecutar scripts PHP, ha demostrado ser una arquitectura coherente, robusta y alineada con buenas prácticas modernas de desarrollo web y despliegue en la nube.

---

---

## 10. Instalación, ejecución, configuración y puesta en marcha

Para que la aplicación "LagVis" funcione correctamente, es necesario seguir una serie de pasos que incluyen la preparación del entorno, la configuración de la base de datos y el despliegue del servicio backend. Esta guía detallada cubre los requisitos del sistema y las instrucciones para cada componente.

### 10.1. Requisitos del Sistema

Para poder instalar, configurar y ejecutar la aplicación, se necesitan los siguientes requisitos de hardware y software:

- **Requisitos de Hardware:**

- **Ordenador de Desarrollo:** Un PC o portátil con al menos 8 GB de RAM (se recomienda 16 GB o más para un rendimiento óptimo), un procesador Intel Core i5 o AMD Ryzen 5 (o superior) y un SSD (disco de estado sólido) para una mayor velocidad de carga y compilación.
- **Smartphone Android:** Un dispositivo Android físico para pruebas funcionales y validación del diseño en un entorno real. Es fundamental para evaluar la experiencia de usuario y la compatibilidad con diferentes versiones de Android y tamaños de pantalla.

- **Requisitos de Software:**

- **Sistema Operativo:** Windows, macOS o Linux (cualquiera compatible con Android Studio).
- **Android Studio:** La última versión estable de Android Studio instalada en el ordenador de desarrollo. Es la herramienta oficial para crear aplicaciones Android y compatible con el lenguaje Java.
- **Java Development Kit (JDK):** Versión compatible con Android Studio.
- **SDK de Android:** Componentes SDK necesarios instalados a través de Android Studio (API de Android, herramientas de construcción, etc.).
- **Docker Desktop:** Instalado en el ordenador de desarrollo para la gestión y construcción de imágenes Docker, aunque el despliegue final se hará en Cloud Run.
- **Google Cloud SDK (gcloud CLI):** Herramienta de línea de comandos para interactuar con los servicios de Google Cloud (Cloud SQL, Cloud Run). Esto es clave para gestionar la base de datos y el backend.
- **Git:** Para clonar el repositorio del proyecto desde GitHub.

- **Navegador Web:** Para acceder a la consola de Google Cloud Platform.

## 10.2. Pasos para la Instalación

La instalación de la aplicación implica la configuración de la base de datos, el despliegue del backend y la compilación de la aplicación móvil.

### 10.2.1. Configuración de la Base de Datos (Google Cloud SQL - MySQL)

#### 1. Crear un Proyecto en Google Cloud Platform (GCP):

- Acceder a la consola de Google Cloud Platform ([console.cloud.google.com](https://console.cloud.google.com)).
- Crear un nuevo proyecto o seleccionar uno existente.

#### 2. Habilitar la API de Cloud SQL:

- Desde el panel de GCP, buscar "SQL" y habilitar la API de Cloud SQL para el proyecto.

#### 3. Crear una Instancia de Cloud SQL (MySQL):

- En el menú de navegación, ir a Bases de datos SQL -> Crear instancia.
- Seleccionar **MySQL** como motor de base de datos.
- Configurar los parámetros de la instancia (nombre, contraseña de usuario *root*, región, tipo de máquina, almacenamiento). Se recomienda elegir una región cercana para minimizar la latencia.
- Configurar la conectividad: Asegurarse de que la instancia permita conexiones desde Cloud Run y Redes autorizadas si se va a acceder desde IPs externas para desarrollo o pruebas.

#### 4. Crear la Base de Datos y las Tablas:

- Una vez creada la instancia de Cloud SQL, conectarse a ella usando un cliente MySQL (por ejemplo, MySQL Workbench, DBeaver) o la propia consola de Cloud Shell de GCP.
- Crear la base de datos específica para la aplicación (ej. lagvis\_db).
- Ejecutar los scripts SQL proporcionados en el repositorio del proyecto para crear las tablas USUARIOS, CONVENIOS y NOTICIAS con sus respectivos campos.
- Insertar datos iniciales o de prueba si es necesario.

### 10.2.2. Despliegue del Backend (Google Cloud Run con Docker)

#### 1. Construir la Imagen Docker del Backend:

- Navegar al directorio del proyecto backend (donde se encuentre el Dockerfile y el código del backend, por ejemplo, en Java).
- Desde la línea de comandos, construir la imagen Docker. Por ejemplo:

Bash

```
docker build -t gcr.io/[PROJECT_ID]/lagvis-backend:latest .
```

(Reemplazar [PROJECT\_ID] por el ID de tu proyecto de Google Cloud).

#### 2. Subir la Imagen Docker a Google Container Registry (GCR):

- Autenticarse con Google Cloud:

Bash

```
gcloud auth configure-docker
```

- Subir la imagen:

Bash

```
docker push gcr.io/[PROJECT_ID]/lagvis-backend:latest
```

#### 3. Desplegar el Contenedor en Google Cloud Run:

- Utilizar el comando gcloud run deploy para desplegar el servicio. Es crucial configurar las variables de entorno para la conexión a la base de datos.
- Ejemplo de comando de despliegue:

Bash

```
gcloud run deploy lagvis-backend \
--image gcr.io/[PROJECT_ID]/lagvis-backend:latest \
--platform managed \
--region [REGION] \
--allow-unauthenticated \ # O configurar la autenticación de
servicio si es necesario
```

---

---

```
--add-cloudsql-instances [PROJECT_ID]:[REGION]:[CLOUD_SQL_INSTANCE_NAME] \
--set-env-vars DB_USER=[MYSQL_USER],DB_PASSWORD=[MYSQL_PASSWORD],DB_NAME=[MYSQL_DB_NAME],DB_HOST=/cloudsql/[PROJECT_ID]:[REGION]:[CLOUD_SQL_INSTANCE_NAME]
(Reemplazar [PROJECT_ID], [REGION], [CLOUD_SQL_INSTANCE_NAME], [MYSQL_USER], [MYSQL_PASSWORD], [MYSQL_DB_NAME] con tus datos reales).
```

- Este comando asocia el servicio Cloud Run con la instancia de Cloud SQL, permitiendo una conexión segura y privada.

#### 10.2.3. Instalación de la Aplicación Móvil (Android Studio)

##### 1. Clonar el Repositorio del Proyecto:

- En el ordenador de desarrollo, abrir una terminal y ejecutar:

Bash

```
git clone https://github.com/SantiagoPerezRu/LagVis.git
```

##### 2. Abrir el Proyecto en Android Studio:

- Abrir Android Studio.
- Seleccionar Open an existing Android Studio project y navegar a la carpeta donde se clonó el proyecto LagVis.
- Esperar a que Android Studio sincronice el proyecto con Gradle y descargue las dependencias necesarias. Esto puede tardar unos minutos la primera vez.

### 10.3. Configuración Inicial

Una vez instalados los componentes, se deben realizar las configuraciones iniciales para que la aplicación móvil se comunique correctamente con el backend y las bases de datos.

- **Configuración en la Aplicación Android (Java):**

- Localizar el archivo de configuración LagVisConstantes.java
- Actualizar la URL base del backend de Cloud Run con la URL obtenida tras el despliegue en el paso 10.2.2.3. Esta URL es el "endpoint" al que la aplicación móvil realizará las llamadas HTTP.
- Verificar las claves de Firebase (si están configuradas en el proyecto Android), aunque Android Studio suele gestionar esto automáticamente si el proyecto está vinculado a Firebase.

- **Archivos .env o Variables de Entorno del Backend:**

- Confirmar que las variables de entorno (DB\_USER, DB\_PASSWORD, DB\_NAME, DB\_HOST) configuradas durante el despliegue de Cloud Run son correctas y coinciden con las credenciales de tu instancia de Google Cloud SQL. Estas variables son cruciales para que el backend se conecte a la base de datos MySQL.

---

### 10.4. Ejecución y Puesta en Marcha

Con todos los componentes configurados, la aplicación está lista para ser ejecutada y puesta en marcha.

1. **Iniciar la Base de Datos (Cloud SQL):**

- La instancia de Google Cloud SQL está diseñada para estar siempre activa una vez desplegada. No se requieren comandos adicionales para "levantarla".

2. **Iniciar el Backend (Cloud Run):**

- El servicio en Google Cloud Run también se gestiona automáticamente. Se activa y escala según la demanda, por lo que no hay un comando de "inicio" manual. Simplemente, estará disponible en la URL que se obtuvo durante el despliegue.

### 3. Ejecutar la Aplicación Móvil (Android Studio):

- En Android Studio, asegúrate de tener un emulador Android configurado o de tener un dispositivo Android físico conectado al ordenador (y con la depuración USB activada).
- Hacer clic en el botón Run 'app' (el icono de triángulo verde) en la barra de herramientas de Android Studio.
- Seleccionar el dispositivo o emulador donde se desea ejecutar la aplicación.
- Android Studio compilará la aplicación (si no se ha hecho ya) y la instalará en el dispositivo seleccionado.
- Una vez instalada, la aplicación se iniciará automáticamente.

Tras estos pasos, la aplicación "LagVis" estará completamente operativa, permitiendo a los usuarios registrarse, buscar convenios, ver noticias y utilizar las calculadoras, interactuando con el backend desplegado en Google Cloud Run y la base de datos MySQL en Google Cloud SQL.

## 11. Conclusiones

Llegar a este punto, con la aplicación "LagVis" ya terminada, me permite echar la vista atrás y ver todo el camino que he recorrido. Es el momento de evaluar si he cumplido todo lo que me propuse al principio, de hablar de las dificultades que encontré y de reflexionar sobre todo lo que he aprendido.

### 14.1. Objetivos Alcanzados y No Alcanzados

Cuando empecé, me planteé unos objetivos claros, tanto generales como más específicos, que me sirvieron de guía durante todo el desarrollo de la aplicación.

#### Objetivo General Principal Cumplido:

El gran objetivo era **crear una aplicación móvil que ayudara a trabajadores y jefes en España a entender fácilmente sus derechos y los convenios colectivos, haciendo todo más claro y transparente**. Y sí, puedo decir que "LagVis" lo consigue por completo. Ofrece una plataforma sencilla e intuitiva para consultar toda esa información laboral que antes era tan complicada de encontrar o entender. La necesidad de resolver la falta de conocimiento sobre derechos laborales y facilitar el acceso a la normativa pertinente se ha abordado con éxito.

---

---

### Objetivos Específicos Cumplidos:

La mayoría de los objetivos más concretos que me marqué al principio los he logrado con la finalización de la aplicación:

- **Crear un sistema seguro para que los usuarios se registren y puedan entrar.** Esto está funcionando perfectamente gracias a Firebase Authentication. Cada usuario tiene su cuenta personalizada y segura.
  - **Diseñar y poner en marcha un buscador de convenios colectivos eficiente, que permita filtrar por el tipo de trabajo y la ubicación.** Esto ha sido clave. La app permite buscar convenios filtrando por comunidad autónoma y sector laboral.
  - **Desarrollar una sección donde se muestren los convenios de forma resumida, con datos clave como el salario mínimo o los días libres.** La aplicación consigue presentar esa información tan densa de forma clara y fácil de entender, lo cual era fundamental.
  - **Integrar una sección con noticias laborales actualizadas sobre cambios en leyes y normativas.** La sección de noticias ya está ahí, mostrando información relevante para que los usuarios estén al día.
  - **Diseñar una interfaz de usuario intuitiva que haga la aplicación fácil de usar y navegar.** Creo que he conseguido una interfaz moderna y amigable, pensando siempre en la experiencia del usuario. Los bocetos iniciales y las ideas sobre la UX se han materializado bien.
  - **Hacer pruebas exhaustivas para asegurar que la aplicación funciona bien en distintos móviles Android.** Esta parte es muy importante. He hecho muchas pruebas en diferentes simuladores y en teléfonos reales para asegurarme de que la app es compatible y funciona de forma fluida.
  - **Integrar calculadoras para facilitar gestiones, como la indemnización por despido y el coste de un despido.** Este objetivo, que añadí un poco más tarde, ya está implementado. Ambas calculadoras son funcionales y simplifican mucho las cuentas que antes eran un dolor de cabeza.
  - **Ofrecer acceso directo a herramientas oficiales del estado.** He añadido enlaces o accesos directos a recursos y webs oficiales que pueden ser útiles para los usuarios, lo que le da un valor extra a la aplicación.
  - **¡Un extra que no estaba previsto: Un calendario laboral funcional!** Aunque no lo tenía en el plan inicial del anteproyecto, decidí implementar un calendario laboral dentro de la aplicación. Lo más interesante es que se rige por la comunidad autónoma del usuario, mostrando los festivos y días importantes específicos de cada región. Esto ha sido una mejora de última hora que creo que aporta mucho valor práctico a la app.
-

### Objetivos No Alcanzados:

Hay algunos objetivos que, de momento, no he podido incluir en esta versión final de la aplicación. Es importante aclarar que esto no es un fallo, sino una decisión de priorización para poder entregar el proyecto a tiempo con sus funcionalidades principales ya funcionando:

- **Sistema de notificaciones automatizadas:** Es algo que requiere una implementación más compleja y quedará para una actualización posterior.
- **Portal de abogados:** Aunque no estaba formalmente en mi anteproyecto, sí que estaba en mi mente la idea de integrar un apartado que conectara a los usuarios con abogados o servicios legales especializados. Sabía que era algo muy potente y de gran valor para los usuarios que buscan soporte legal directo, pero también que implicaba una complejidad considerable: no solo a nivel técnico, sino también la gestión de posibles colaboraciones externas, aspectos legales de responsabilidad y la adaptación a un modelo de negocio que excedía el alcance de este Trabajo de Fin de Grado. Por eso, aunque la idea sigue siendo atractiva para el futuro, no ha podido ser implementada en esta primera versión.

### 14.2. Valoración Personal y Dificultades Encontradas

Hacer este Trabajo de Fin de Grado ha sido una experiencia increíblemente enriquecedora y, a la vez, un gran desafío.

**Mi Valoración Personal:** Estoy muy contento con el resultado final. Ver "LagVis" funcionando, sabiendo que cumple con el propósito de ayudar a la gente con un tema tan importante como los derechos laborales, me llena de satisfacción. Ha sido una oportunidad única para poner en práctica todo lo que aprendí en el Grado Superior, sobre todo con Java en Android Studio, y también para adentrarme en el mundo del desarrollo backend y la nube con Google Cloud y Firebase. Pasar de una simple idea a una aplicación real y funcional ha sido una aventura que me ha enseñado muchísimo.

### Las Dificultades que me he encontrado:

Durante el desarrollo, me topé con varios obstáculos, pero cada uno de ellos fue una oportunidad para aprender y mejorar:

- **Entender y estructurar la información de los convenios:** Fue un quebradero de cabeza al principio. Los convenios son muy extensos y complejos. Decidir qué información era la más importante para mostrarla sin que la app fuera un lio fue un desafío. Al final, lo resolví centrándome en lo más vital: salarios, permisos y vacaciones. Esto me obligó a simplificar mucho sin perder la esencia legal.

- **Conectar la aplicación móvil con la base de datos y el backend:** Hacer que la app "hablara" correctamente con el servicio backend en Cloud Run y, a su vez, con la base de datos MySQL en Google Cloud SQL, no fue trivial. Hubo que configurar bien las conexiones, las variables de entorno y asegurarse de que los datos viajaran de forma eficiente y segura. Han sido mucha horas para poder implementar todo el backend en la nube pero ha merecido completamente la pena.
- **Gestionar el tiempo y los recursos como único desarrollador:** Como todo lo he hecho yo solo, tuve que ser muy organizado para repartir mi tiempo entre el diseño, la programación, la configuración de la nube y las pruebas. La metodología Scrum me ayudó, pero mantener el ritmo y la disciplina fue un desafío constante.
- **Aprender a usar herramientas nuevas de Google Cloud:** Aunque ya usaba Firebase, tuve que aprender mucho sobre cómo funcionan Google Cloud SQL y, especialmente, Cloud Run con Docker. Entender cómo subir contenedores, cómo se conectan con la base de datos, etc., me llevó tiempo. Pero al final, me di cuenta de lo potentes que son y lo mucho que me han servido para este proyecto.

En resumen, las dificultades fueron parte del viaje de aprendizaje. Cada problema que logré superar me hizo más fuerte en mis habilidades técnicas y en mi capacidad para resolver retos por mí mismo. Todo esto ha culminado en la entrega exitosa de una aplicación funcional y relevante para el mundo laboral en España.

## 12. Vías Futuras

Aunque la aplicación "LagVis" ya es funcional y cumple con los objetivos principales, siempre hay margen para mejorar y añadir nuevas funcionalidades que la hagan aún más completa y útil.

### Possibles Mejoras y Funcionalidades Futuras:

En el futuro, me gustaría explorar las siguientes mejoras para la aplicación:

- **Sistema de Notificaciones Automatizadas:** Como ya comenté en las conclusiones, la infraestructura para las notificaciones está preparada, pero sería muy interesante implementar la lógica para enviar alertas personalizadas a los usuarios. Por ejemplo, notificaciones cuando haya cambios en sus convenios colectivos, nuevas noticias relevantes para su sector, o incluso recordatorios sobre plazos importantes.

- **Portal de Abogados:** Aunque no fue posible incluirlo en esta primera versión, sigo creyendo que un directorio de abogados especializados en derecho laboral sería un valor añadido enorme para la aplicación. Permitiría a los usuarios contactar directamente con profesionales para resolver dudas o recibir asesoramiento legal.
- **Pasarelas de Pago:** Si en el futuro se decide ofrecer servicios premium dentro de la aplicación (por ejemplo, acceso a contenido exclusivo o herramientas avanzadas), sería necesario integrar pasarelas de pago seguras y fiables para gestionar las transacciones.

#### **Migración a Flutter para Desarrollo Multiplataforma:**

Una de las vías futuras más interesantes para "LagVis" sería **migrar el desarrollo de la aplicación a Flutter**. Actualmente, la aplicación está desarrollada en Java para Android, lo que significa que solo funciona en dispositivos Android. Flutter, un framework de desarrollo de Google, ofrece una serie de ventajas significativas frente a otros frameworks multiplataforma (como React Native o Xamarin) que lo hacen ideal para el futuro de "LagVis":

- **Un Único Código Base para Múltiples Plataformas:** La principal ventaja de Flutter es que permite desarrollar aplicaciones para iOS y Android (e incluso web y escritorio) **con un único código base**. Esto significa que no sería necesario mantener dos proyectos separados (uno para Android y otro para iOS), lo que reduce drásticamente el tiempo de desarrollo, los costes de mantenimiento y la complejidad del proyecto.
- **Rendimiento Nativo:** Flutter compila directamente a código nativo, lo que se traduce en un **rendimiento muy similar al de las aplicaciones desarrolladas con lenguajes nativos** (Java/Kotlin para Android y Swift para iOS). Esto es crucial para una aplicación como "LagVis", donde la fluidez y la rapidez son importantes para la experiencia del usuario. Otros frameworks multiplataforma suelen utilizar "puentes" entre el código JavaScript y el código nativo, lo que puede generar una ligera pérdida de rendimiento.
- **Desarrollo Rápido ("Hot Reload"):** Flutter ofrece una funcionalidad llamada "Hot Reload" que permite **ver los cambios en el código casi al instante en la aplicación en ejecución**, sin necesidad de recompilarla por completo. Esto acelera enormemente el proceso de desarrollo y facilita la experimentación con la interfaz de usuario.
- **Interfaz de Usuario Expresiva y Personalizable:** Flutter ofrece una **gran cantidad de widgets (componentes de la interfaz de usuario)** **altamente personalizables**, lo que permite crear interfaces de usuario modernas, atractivas y adaptadas a la identidad visual de "LagVis". Además, Flutter tiene un excelente soporte para animaciones y transiciones, lo que permite crear experiencias de usuario más ricas e interactivas.

**En resumen:** Migrar "LagVis" a Flutter permitiría **ampliar la base de usuarios al llegar también a dispositivos iOS, reducir los costes de desarrollo y mantenimiento a largo plazo, y mejorar la experiencia del usuario con una interfaz más moderna y un rendimiento nativo**. Esta decisión estratégica posicionaría a "LagVis" para un futuro aún más exitoso.

### 13. Glosario:

**Android Studio:** Es el **Entorno de Desarrollo Integrado (IDE)** oficial proporcionado por Google para el desarrollo de aplicaciones para el sistema operativo Android. Funciona como un "taller" o programa que concentra todas las herramientas necesarias para escribir, depurar y compilar código para Android.

**API (Application Programming Interface):** Conjunto de definiciones y protocolos que se utiliza para diseñar y construir software de aplicación. En este contexto, se refiere a la forma en que los diferentes componentes de la aplicación (como la app móvil y el backend) se comunican entre sí.

**Autenticación:** Proceso de verificar la identidad de un usuario, generalmente mediante un nombre de usuario y una contraseña. En la aplicación, permite que los usuarios se registren e inicien sesión de forma segura.

**Backend:** Se refiere a la parte "del lado del servidor" de una aplicación, que no es visible directamente para el usuario final. Incluye la lógica de negocio, la base de datos y la comunicación con el servidor. En este proyecto, el backend se gestiona con Google Cloud Run y se conecta a Google Cloud SQL.

**Balsamiq Mockups:** Herramienta de software utilizada para crear **bocetos iniciales** o "wireframes" de interfaces de usuario. Ayuda a visualizar la estructura y el flujo de una aplicación antes de programarla.

**Base de Datos (BBDD):** Conjunto organizado de datos que se almacena y se recupera de manera eficiente. En este proyecto, se utilizan tanto Firebase como Google Cloud SQL (MySQL).

**Cloud Run:** Servicio de Google Cloud que permite ejecutar contenedores (aplicaciones empaquetadas) sin necesidad de gestionar servidores. Escala automáticamente según la demanda y solo se paga por el tiempo de uso del procesador, optimizando costes.

**Convenios Colectivos:** Acuerdos entre los representantes de los trabajadores y los empresarios que regulan las condiciones de trabajo en un sector o empresa. La aplicación busca facilitar su comprensión.

---

---

**CRUD:** Acrónimo de **Create, Read, Update, Delete** (Crear, Leer, Actualizar, Borrar). Se refiere a las cuatro operaciones básicas que se pueden realizar sobre los datos almacenados en una base de datos.

**Depuración:** Proceso de identificar y corregir errores (bugs) en el código de un programa.

**Docker:** Plataforma que permite empaquetar una aplicación y todas sus dependencias (librerías, configuraciones) en un "contenedor" estandarizado. Esto asegura que la aplicación funcione de la misma manera en cualquier entorno.

**Dockerfile:** Archivo de texto que contiene las instrucciones para construir una imagen Docker.

**Framework:** Conjunto de herramientas y librerías predefinidas que proporcionan una estructura base para el desarrollo de software. Facilita la creación de aplicaciones al ofrecer soluciones comunes a problemas recurrentes.

**Firebase:** Plataforma de desarrollo de Google que ofrece servicios de backend, como bases de datos en tiempo real (NoSQL) y autenticación de usuarios, de forma fácil de integrar.

**Flutter:** Framework de desarrollo de UI de código abierto creado por Google para construir aplicaciones compiladas de forma nativa para móvil, web y escritorio a partir de una única base de código.

**Frontend:** Se refiere a la parte de una aplicación con la que el usuario interactúa directamente (la interfaz de usuario). En este proyecto, la aplicación móvil desarrollada con Android Studio y Java es el frontend.

**Gantt (Diagrama de Gantt):** Herramienta gráfica utilizada en la gestión de proyectos para representar el cronograma de tareas. Muestra las fechas de inicio y fin, así como la duración de cada actividad.

**gcloud CLI:** (Google Cloud SDK Command-Line Interface) Es la herramienta de línea de comandos proporcionada por Google para interactuar con los servicios de Google Cloud Platform.

**GitHub:** Plataforma de control de versiones basada en Git que permite a los desarrolladores alojar y gestionar sus proyectos de software. Facilita el seguimiento de cambios y la colaboración.

**Google Cloud Platform (GCP):** Conjunto de servicios de computación en la nube ofrecidos por Google. Incluye infraestructura, plataformas y soluciones para desarrollo de software.

**Google Cloud SQL:** Servicio de base de datos relacional completamente gestionado por Google Cloud, compatible con MySQL, PostgreSQL y SQL Server.

---

---

**Hot Reload:** Característica de Flutter que permite a los desarrolladores ver los cambios en el código reflejados en la aplicación en ejecución casi instantáneamente, sin necesidad de recompilarla por completo.

**IDE (Integrated Development Environment):** Un entorno de desarrollo integrado es un software que proporciona herramientas integrales a los programadores para el desarrollo de software, como un editor de código, un depurador y un compilador. Android Studio es un IDE.

**Java:** Lenguaje de programación orientado a objetos, ampliamente utilizado para desarrollar aplicaciones móviles Android, aplicaciones empresariales y web.

**JDK (Java Development Kit):** Kit de desarrollo de software para aplicaciones Java, esencial para compilar y ejecutar código Java.

**Kanban:** Metodología ágil que visualiza el flujo de trabajo mediante tableros y tarjetas, permitiendo un seguimiento del progreso y la identificación de cuellos de botella.

**Layout:** En el contexto de Android, se refiere a la estructura visual de una interfaz de usuario, definiendo cómo se organizan los elementos (botones, textos, imágenes) en una pantalla. Se define comúnmente con XML.

- **Licencia Propietaria Comercial:** Tipo de licencia de software que restringe el uso, la modificación, la copia y la redistribución del software, generalmente exigiendo un pago por su uso comercial o redistribución.
- **MySQL:** Sistema de gestión de bases de datos relacional de código abierto, muy popular y robusto.

**Scrum:** Metodología ágil de desarrollo de software que organiza el trabajo en ciclos cortos y definidos llamados "sprints". Promueve la adaptación a cambios y la colaboración constante.

**(Software Development Kit):** Conjunto de herramientas de software que permite a los desarrolladores crear aplicaciones para una plataforma específica (ej. Android SDK).

**Serverless:** Modelo de ejecución en la nube donde el proveedor de la nube gestiona la infraestructura del servidor, permitiendo a los desarrolladores escribir y desplegar código sin preocuparse por la gestión de servidores. Google Cloud Run es un servicio serverless.

**Sprint:** Periodo de tiempo fijo y corto (en este caso, 2 semanas) dentro de la metodología Scrum, durante el cual se realiza un conjunto de tareas planificadas.

**SQL (Structured Query Language):** Lenguaje estándar utilizado para comunicarse con bases de datos relacionales, permitiendo gestionar y manipular los datos.

---

**UI (User Interface):** Interfaz de Usuario. Se refiere a la parte visual de una aplicación con la que el usuario interactúa.

**UX (User Experience):** Experiencia de Usuario. Se refiere a cómo se siente un usuario al interactuar con una aplicación, incluyendo su facilidad de uso, eficiencia y satisfacción.

**XML (eXtensible Markup Language):** Lenguaje de marcado utilizado para codificar documentos en un formato legible tanto para humanos como para máquinas. En Android, se utiliza para definir los layouts de las interfaces de usuario.

## 14. WEBGRAFÍA

<https://developer.android.com/?hl=es-419>

<https://docs.oracle.com/en/java/javase/17/docs/api/>

<https://cloud.google.com/docs?hl=es-419>

<https://docs.docker.com/>

<https://www.php.net/docs.php>

Anexos:

Código fuente en GitHub: <https://github.com/SantiagoPerezRu/LagVis>

Código fuente en Google Drive: [Descargar esta versión!](#)